Contents

Explicit Deny

4 minute read page test

This task shows you how to set up Istio authorization policy of `DENY` action to explicitly deny traffic in an Istio mesh. This is different from the `ALLOW` action because the `DENY` action has higher priority and will not be bypassed by any `ALLOW` actions.

Before you begin

Before you begin this task, do the following:

- Read the Istio authorization concepts.
- Follow the Istio installation guide to install Istio.
- Deploy workloads:

  This task uses two workloads, httpbin and sleep, deployed on one namespace, foo. Both workloads run with an Envoy proxy in front of each. Deploy the example namespace and workloads with the following command:

```
$ kubectl create ns foo
$ kubectl apply -f <(istioctl kube-inject -f @samples/httpb
in/httpbin.yaml@) -n foo
$ kubectl apply -f <(istioctl kube-inject -f @samples/sleep
/sleep.yaml@) -n foo
```

- Verify that `sleep` talks to `httpbin` with the following command:

```
$ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o jsonpat
h={.items..metadata.name})" -c sleep -n foo -- curl http://httpb
in.foo:8000/ip -sS -o /dev/null -w "%{http_code}\n"
200
```

If you don’t see the expected output as you follow the task, retry after a few seconds.

Caching and propagation overhead can cause some delay.

Explicitly deny a request

1. The following command creates the `deny-method-get` authorization policy for the `httpbin` workload in the `foo` namespace. The policy sets the `action` to `DENY` to deny requests that satisfy the conditions set in the `rules` section. This type of policy is

better known as deny policy. In this case, the policy denies requests if their method is `GET`.

```
$ kubectl apply -f - <<EOF
apiVersion: security.istio.io/v1beta1
kind: AuthorizationPolicy
metadata:
  name: deny-method-get
  namespace: foo
spec:
  selector:
    matchLabels:
      app: httpbin
  action: DENY
  rules:
  - to:
    - operation:
        methods: ["GET"]
EOF
```

2. Verify that `GET` requests are denied:

```
$ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o js
onpath={.items..metadata.name})" -c sleep -n foo -- curl "h
ttp://httpbin.foo:8000/get" -X GET -sS -o /dev/null -w "%{h
ttp_code}\n"
403
```

3. Verify that `POST` requests are allowed:

```
$ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o js
onpath={.items..metadata.name})" -c sleep -n foo -- curl "h
ttp://httpbin.foo:8000/post" -X POST -sS -o /dev/null -w "%
{http_code}\n"
200
```

4. Update the `deny-method-get` authorization policy to deny `GET` requests only if the value of the HTTP

header `x-token` value is not `admin`. The following example policy sets the value of the `notValues` field to `["admin"]` to deny requests with a header value that is not `admin`:

```
$ kubectl apply -f - <<EOF
apiVersion: security.istio.io/v1beta1
kind: AuthorizationPolicy
metadata:
  name: deny-method-get
  namespace: foo
spec:
  selector:
    matchLabels:
      app: httpbin
  action: DENY
  rules:
  - to:
    - operation:
        methods: ["GET"]
    when:
    - key: request.headers[x-token]
      notValues: ["admin"]
EOF
```

5. Verify that `GET` requests with the HTTP header `x-token: admin` are allowed:

   ```
   $ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o js
   onpath={.items..metadata.name})" -c sleep -n foo -- curl "h
   ttp://httpbin.foo:8000/get" -X GET -H "x-token: admin" -sS
   -o /dev/null -w "%{http_code}\n"
   200
   ```

6. Verify that GET requests with the HTTP header `x-token: guest` are denied:

   ```
   $ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o js
   onpath={.items..metadata.name})" -c sleep -n foo -- curl "h
   ttp://httpbin.foo:8000/get" -X GET -H "x-token: guest" -sS
   -o /dev/null -w "%{http_code}\n"
   403
   ```

7. The following command creates the `allow-path-ip` authorization policy to allow requests at the `/ip` path to the `httpbin` workload. This authorization policy sets the `action` field to `ALLOW`. This type of policy is better known as an allow policy.

```
$ kubectl apply -f - <<EOF
apiVersion: security.istio.io/v1beta1
kind: AuthorizationPolicy
metadata:
  name: allow-path-ip
  namespace: foo
spec:
  selector:
    matchLabels:
      app: httpbin
  action: ALLOW
  rules:
  - to:
    - operation:
        paths: ["/ip"]
EOF
```

8. Verify that `GET` requests with the HTTP header `x-token: guest` at path `/ip` are denied by the `deny-`

`method-get` policy. Deny policies takes precedence over the allow policies:

```
$ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o js
onpath={.items..metadata.name})" -c sleep -n foo -- curl "h
ttp://httpbin.foo:8000/ip" -X GET -H "x-token: guest" -s -o
 /dev/null -w "%{http_code}\n"
403
```

9. Verify that `GET` requests with the HTTP header `x-token: admin` at path `/ip` are allowed by the `allow-path-ip` policy:

```
$ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o js
onpath={.items..metadata.name})" -c sleep -n foo -- curl "h
ttp://httpbin.foo:8000/ip" -X GET -H "x-token: admin" -s -o
 /dev/null -w "%{http_code}\n"
200
```

10. Verify that `GET` requests with the HTTP header `x-token: admin` at path `/get` are denied because they don't match the `allow-path-ip` policy:

```
$ kubectl exec "$(kubectl get pod -l app=sleep -n foo -o js
onpath={.items..metadata.name})" -c sleep -n foo -- curl "h
ttp://httpbin.foo:8000/get" -X GET -H "x-token: admin" -s -
o /dev/null -w "%{http_code}\n"
403
```

Clean up

1. Remove the namespace foo from your configuration:

   ```
   $ kubectl delete namespace foo
   ```